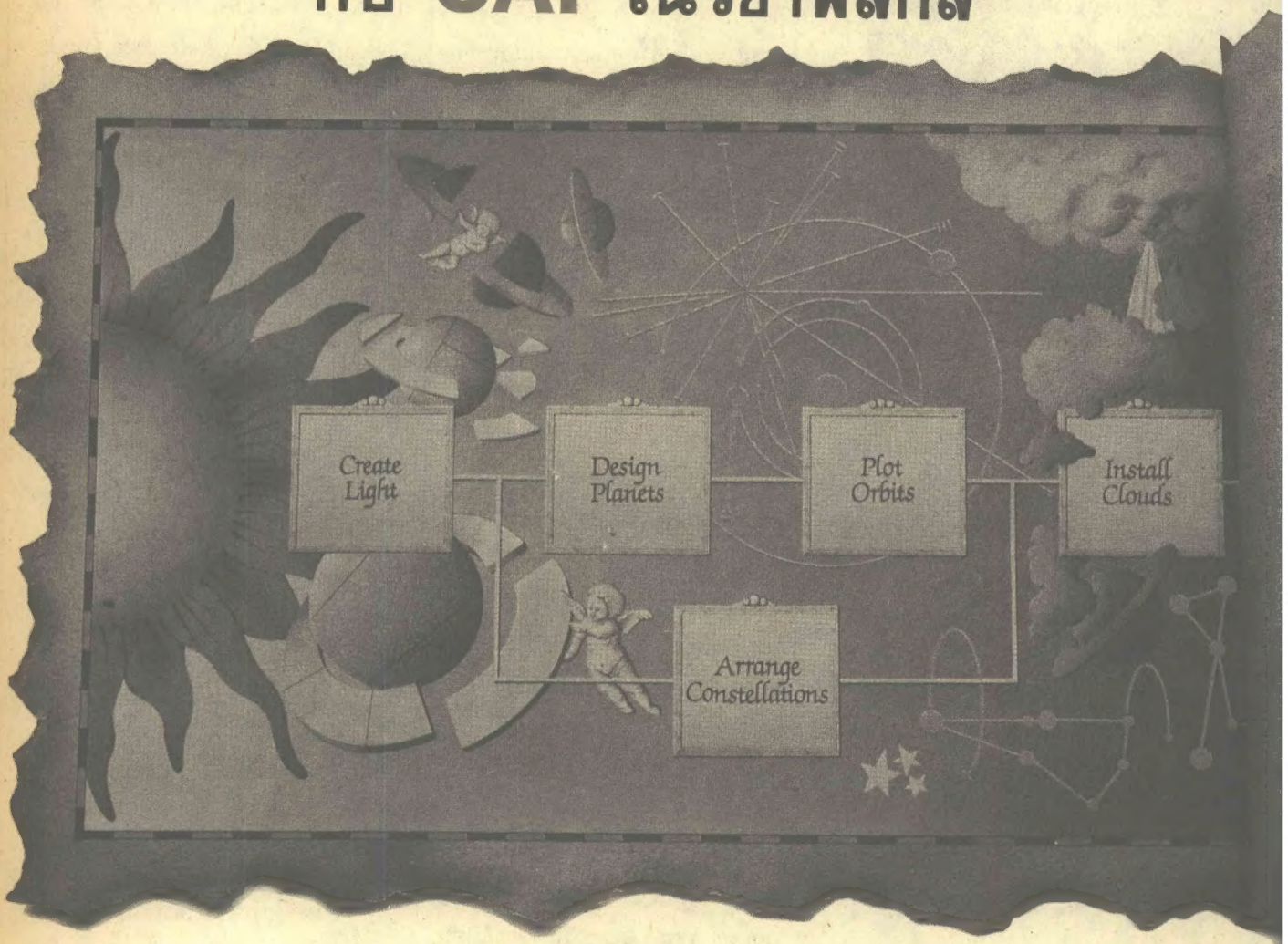
การประยุกต์ใช้เทอร์โบปาสคาล 4.0 กับ CAI ในวิชาฟิสิกส์



ตอน2

อาจหาญ สัตยารักษ์

จากบทความเรื่อง เทอร์โบปาสคาล 4.0 กับ CAI ตอนที่ 1 (ในวารสารไมโครฯ ฉบับที่ 45) นั้นเราได้ศึกษาเกี่ยวกับคำสั่งด้านกราฟิกไปแล้วดังนี้

1. InitGraph(var GraphDrive : integer; var GraphMode : integer; DriverPath : string)
   คำสั่งเพื่อเข้ากราฟิกตามตัวแปรที่กำหนดไว้
2. Set TextStyle(Font : word; Direction : word; CharSize : CharSizeType)
   คำสั่งสำหรับเลือกรูปแบบตัวอักษร, การแสดงผลแนวนอนหรือแนวดิ่ง และขนาด
3. OutTextXY(X,Y : integer; TextString :string)
   คำสั่งใช้ในการแสดงข้อความที่กำหนด ณ จุด X,Y ที่ระบุในคำสั่งนี้
4. Ellipse(X,Y :integer; StAngle,EnAngle :word; XRadius, YRadius : word)
   คำสั่งสำหรับวาดวงรีเต็มวงหรือส่วนหนึ่งของวงรีก็ได้
5. Line(X1,Y1,X2,Y2 : integer)
   คำสั่งสำหรับลากเส้นตรง

6. MoveTo(X,Y : integer)
คำสั่งในการเลื่อนตำแหน่งเริ่มต้นของการลงจุดกราฟิก

7. LineTo(X,Y : integer)
คำสั่งลากเส้นจากจุดเดิมไปยังจุดที่กำหนด

8. CloseGraph
คำสั่งเพื่อออกจากกราฟิก

รวมทั้งสิ้น 8 คำสั่ง พร้อมกับได้สร้างคำสั่งใหม่ขึ้นมาใช้อีก 3 คำสั่งคือ

OpenGraph เพื่อเข้าสู่กราฟิก

WaitToGo ใช้เมื่อต้องการรอกดแป้น หรือหยุดภาพชั่วขณะ

TextShow ใช้งานลักษณะเดียวกันกับ SetTextStyle แต่กำหนดรูปแบบง่ายกว่า

โดยผู้เขียนได้เสนอแนะให้เก็บโปรแกรมจากรูปที่ 1 ไว้ใช้งานในชื่อ INITGR.PAS เพื่อเรียกมาใช้ในภายหลังด้วยการกด ctrl-K R และป้อนชื่อไฟล์นี้แบบหนึ่ง หรือใช้คำสั่งให้ include โปรแกรมนี้เข้ามาด้วยคำสั่ง {SI INITGR.PAS} ที่ต้นโปรแกรมซึ่งเราพัฒนาในครั้งต่อไป ดังจะกล่าวถึงในตอนที่ 2 นี้เป็นต้น

เรามาศึกษาคำสั่งของเทอร์โบปาสคาล 4.0 เพิ่มเติมกันต่อ ซึ่งมีทั้งคำสั่งด้านกราฟิกและคำสั่งด้านอื่น ดังนี้

1. SetColor(color :word) คือคำสั่งในการกำหนดสีในการสร้างกราฟิกสำหรับการทำบนจอโมโนโครมก็จะมีเพียง 2 สี คือ 0,1 เช่น SetColor(1); หรือ SetColor(0);

2. SetFillStyle(Pattern :word; Color :word) คือคำสั่งในการเลือกรูปแบบที่จะใช้ระบายลงในกล่องทึบ 11 แบบ (ค่า pattern มี 0-10) เช่น SetFillStyle(4,1);

3. FloodFill(X,Y,Border : word) คำสั่งเมื่อต้องการระบายลงในรูปทึบ จุด X,Y ต้องอยู่ในรูปกล่องทึบปิดเสมอ มิฉะนั้นการระบายสีจะเลอะออกมาข้างนอกกรอบได้ ตัวอย่างคือ

FloodFill(200,50,blue);

4. Str(X [:width [:decimals]], var s : string,] คำสั่งเปลี่ยนตัวเลขให้เป็นอักขระ เหมือนกับในเทอร์โบปาสคาล 3.0 เช่น Str(123,s);

5. Bar(X1,Y1,X2,Y2 : integer) คำสั่งเพื่อวาดกล่องสีเหลี่ยม ซึ่งจะเป็นกล่องโปร่งหรือกล่องทึบแบบต่าง ๆ ก็ได้ (ถ้าใช้ร่วมกับคำสั่ง SetFillStyle) โดย X1,Y1 เป็นมุมบนซ้ายส่วน X2,Y2 เป็นมุมล่างขวา เช่น Bar(10,20,300,150);

6. Circle(X,Y : integer; Radius : word) คำสั่งในการวาดวงกลม ณ จุดศูนย์กลาง X,Y โดยมีรัศมีเท่ากับค่า Radius เช่น Circle(200,100,50);

7. Arc(X,Y : integer; StAngle,EnAngle,Radius : word) คำสั่งในการวาดส่วนโค้งของวงกลม เช่น Arc(200, 100,0,90,40); ค่า StAngle,EnAngle ซึ่งเป็นมุมเริ่มต้นและมุมสิ้นสุดของส่วนโค้งนั้นเป็นองศา

8. ImageSize(X1,Y1,X2,Y2 : word) คำสั่งกำหนดขอบเขตของรูปที่จะให้จองเนื้อที่ในหน่วยความจำเพื่อเก็บรูปดังกล่าวนี้ในตัวแปรที่กำหนด(ใช้ร่วมกับคำสั่งที่ 9,10,11,)

9. GetMem(var p :pointer; Size : word) คำสั่งเพื่อจองเนื้อที่ในหน่วยความจำให้แก่ตัวแปรที่จะจำรูปใดรูปหนึ่งเก็บไว้ p คือชื่อของตัวแปร, Size คือขนาดของรูปที่กำหนดตามคำสั่งที่ 8 ข้างต้น

10. GetImage(X1,Y1,X2,Y2 : word; var BitMap) คำสั่งให้จำรูปที่ได้กำหนดไว้นั้นลงในหน่วยความจำที่จองเนื้อที่แล้ว หมายถึงว่าค่าของ X1,Y1,X2,Y2 ต้องเป็นค่าเดียวกับที่กำหนดไว้ในคำสั่งที่ 8 เช่น GetImage (20,30,50,80,p^); โปรดสังเกต BitMap คือชื่อตัวแปร p ที่กำหนดไว้เดิมในคำสั่งที่ 9 แต่คราวนี้ต้องเติมด้วย ^ให้เป็น p^

11. PutImage (X1, Y1 : integer; Var BitMap; BitBit:word) คำสั่งที่นำรูปที่เก็บไว้ด้วยคำสั่งที่ 10 มาแสดง ณ ตำแหน่งที่กำหนดใหม่ โดยแสดงได้ 5 แบบ(ค่า BitBit)คือ

การแสดงแบบ	ค่าของ BitBit	เช่น
NormalPut	0	PutImage (200,120,0);
XorPut	1	PutImage (200,120,1);
OrPut	2	PutImage (200,120,2);
AndPut	3	PutImage (200,120,3);
NotPut	4	PutImage (200,120,4);

ซึ่งในโปรแกรมที่จะใช้งานต่อไปนี้เป็นการกำหนดแบบ NormalPut ค่า X,Y คือมุมบนซ้ายของรูปที่จะให้ปรากฏ

คำสั่งที่ 8,9,10,11 ทำงานร่วมกัน เช่นต้องการย้ายรูปที่ตำแหน่ง 50,100,150,140 ให้ไปปรากฏที่ตำแหน่ง 300, 150 อาจใช้คำสั่งได้ดังนี้

```
size : = ImageSize (50,100,150,140); GetMem(pic, Size);
```

```
GetImage (50, 100, 150, 140, pic^); putImage (300, 150, pic^, 0);
```

โดยต้องกำหนดตัวแปร size,pic ล่วงหน้าไว้ตามลักษณะของปาสคาลด้วย

ก่อนจะใช้ศึกษาด้านกราฟิกต่อไปขอให้ตรวจสอบว่าแผ่นเทอร์โบเบสิกแผ่นที่ 3 ซึ่งมีไฟล์ชื่อ INITGR.PAS อยู่ในเครื่องอ่านแผ่นบันทึกเรียบร้อยแล้ว เพราะเราต้องใช้แผ่นนี้เสมอเมื่อใช้งานด้านกราฟิก (ในตอนต่อไปผู้เขียนจะไม่กล่าวเน้นในเรื่องนี้แล้ว)

เราลองมาดูหน้าตาของรูปแบบที่มีให้ในการระบายกล่องทึบว่าทั้ง 11 แบบนั้นมีหน้าตาของแต่ละแบบเป็นอย่างไร นั่นคือโปรแกรมในรูปที่ 1 ซึ่งมีการเรียกไฟล์ INITGR.PAS ขึ้นมาก่อนด้วยคำสั่ง {$I INITGR.PAS} ดังกล่าวมาแล้ว ขออธิบายรายละเอียดดังนี้

1. กำหนดตัวแปร

mode เพื่อจะใช้เรียกรูปแบบจากคำสั่ง SetFillStyle

X,Y เพื่อใช้กำหนดตำแหน่งการวาดรูปกล่องโปร่ง

tmode ตัวแปรอักขระที่เปลี่ยนมาจากตัวแปรตัวเลข mode

2. เข้าสู่กราฟิก และเลือกสีหมายเลข 1 ซึ่งเป็นกำหนดให้เกิดรูป และกำหนดค่า Mode, x,y เพื่อนำไปใช้งาน

3. วาดรูปสี่เหลี่ยมผืนผ้า(ส่วนของคำสั่ง moveto, lineto) เลือกรูปแบบของการระบาย (SetFillStyle) พร้อมกับระบายรูปแบบที่เลือกนั้นลงในกรอบสี่เหลี่ยม(FloodFill) ที่วาดไว้ และแสดงตัวเลข(tmode) ให้ทราบว่าเป็นของหมายเลขใด

4. เพิ่มค่าตัวแปร Y ให้มากขึ้นอีก 17 และเพิ่มค่าตัวแปร mode อีก 1 แล้วไปทำซ้ำข้อ 3 จนกว่าค่า mode มากกว่า 10 ซึ่งจะมีผลทำให้เกิดรูปสี่เหลี่ยมแบบต่าง ๆ รวม 11 แบบ ดังปรากฏในรูปที่ 2

5. เมื่อค่า mode = 11 ก็รอการกดแป้นและออกจากกราฟิก

ผู้อ่านสามารถเปลี่ยนคำสั่งตามข้อ 3 ไปเป็นคำสั่งท้ายนี้ได้ คือ

```
SetFillStyle(mode,1); Bar(x,y,x+100,y+15);
```

ซึ่งจะได้ภาพคล้าย ๆ กันแต่ต่างตรงที่จะไม่มีกรอบสี่เหลี่ยมเท่านั้น

```
{ ********************* SetFillStyle ********************* }
{$I INITGR.PAS}
var
 mode,x,y :integer;
 tmode :string[2];
begin
 opengraph;   setcolor(1);
 mode :=0;   x :=10;   y :=0;
  repeat
   moveto(x,y);    lineto(x+100,y); lineto(x+100,y+15);
   lineto(x,y+15); lineto(x,y);
   setfillstyle(mode,1);  floodfill(x+5,y+5,blue);
   str(mode,tmode);       outtextxy(x+110,y+5,tmode);
   y :=y+17;              mode :=mode+1;
  until mode >10;
 waittogo;
 closegraph;
end.
```

รูปที่ 1 โปรแกรมสำหรับตรวจสอบคำสั่ง SetFillStyle และ FloodFill

ต่อไปเราลองมาเขียน CAI วิชาฟิสิกส์กันสักหนึ่งเรื่อง โดยคราวนี้ขอพูดถึงเรื่องของพื้นเอียงลื่น(Smooth Incline) ซึ่งเป็นเรื่อง สภาพสมดุล อันเป็นบทเรียนวิชาฟิสิกส์ในชั้น ม.4 โดยผู้เขียนกำหนดให้มีวัตถุ 2 ชิ้น ชิ้นที่หนึ่ง(m_1)ห้อยอยู่ในแนวดิ่ง ส่วนอีกหนึ่งชิ้น(m_2)วางอยู่บนพื้นเอียง และวัตถุทั้งสองผูกติดกันด้วยเชือกที่คล้องกับรอกลื่นที่มุมบนของระนาบเอียงลื่นนี้

โดยโปรแกรมแบ่งออกเป็น 6 ส่วน คือ

1. ส่วนที่เรียกไฟล์ INITGR.PAS ที่จะช่วยให้เข้าสู่กราฟิกโหมดได้

2. ส่วนของชุดคำสั่ง SmoothIncline วาดรูปเริ่มต้นแสดงระนาบเอียงกับวัตถุทั้งสองชิ้นนั้น

3. ส่วนของชุดคำสั่ง Force สำหรับแสดงแรงต่าง ๆ ที่กระทำต่อวัตถุทั้งสอง

4. ส่วนของชุดคำสั่ง Text แสดงคำอธิบายถึงความสัมพันธ์ของแรงต่าง ๆ

5. ส่วนของชุดคำสั่ง Movepix แสดงการเคลื่อนที่ของวัตถุทั้งสองโดยเป็นการเคลื่อนที่แบบที่วัตถุชิ้นที่ห้อยในแนวดิ่งเคลื่อนที่ลง

6. ส่วนของการแสดงภาพตามลำดับที่ได้กำหนดไว้

โปรแกรมทั้ง 6 ส่วนนี้คือโปรแกรมในรูปที่ 3 นั่นเอง

ขออธิบายรายละเอียดของแต่ละตอนดังนี้

ชุดคำสั่ง SmoothIncline (ไม่นับบรรทัด begin) จะมีการทำงานดังนี้

1. สองบรรทัดแรก(moveto,lineto) เป็นการวาดพื้นเอียง

2. บรรทัดที่สาม(Bar) วาดรูปวัตถุชิ้นที่ห้อยอยู่ในแนวดิ่งซึ่งจะเป็นวัตถุสีทึบ

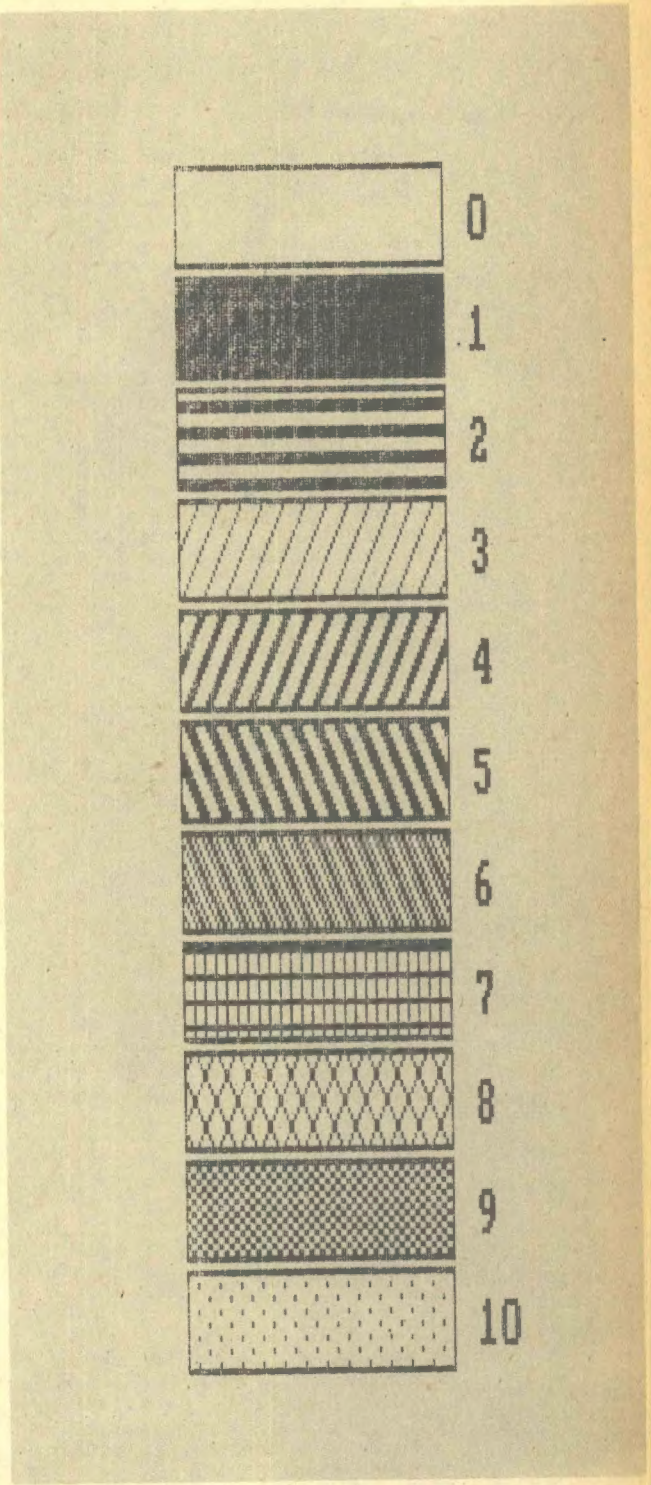
3. บรรทัดที่ 4-6 วาดรูปวัตถุชิ้นที่อยู่บนพื้นเอียง ระบายสีตามรูปแบบที่ 10

4. บรรทัดที่ 7-8 วาดรูปรอกเดี่ยวตายตัว และผูกวัตถุทั้งสองเข้าด้วยกันและคล้องกับรอกดังกล่าว

5. บรรทัดที่ 9-10 แสดงมุม A และเขียนส่วนโค้งจากคำสั่ง arc

ชุดคำสั่ง Force เป็นการเขียนแรงที่กระทำต่อวัตถุพร้อมกับข้อความบอกว่าเป็นแรงชนิดใด โดยมีลักษณะคำสั่งเป็นการเขียนแรงแล้วจบด้วยข้อความแสดงชนิดของแรง ตามลำดับดังนี้ แรง $m_1 g$, แรง T ที่ด้านวัตถุชิ้นที่หนึ่ง,แรง $m_2 g$, แรง $m_2 g \sin A$,แรง $m_2 g \cos A$, แรง R และความเร่ง a ซึ่งจะใช้คำสั่ง moveto,lineto เป็นส่วนใหญ่ โดยชุดคำสั่งนี้มีการกำหนดสี(cc) เพื่อกำหนดให้แสดงแรง,ข้อความเมื่อใช้ Force(1); และลบแรง,ข้อความเมื่อใช้ force(0);

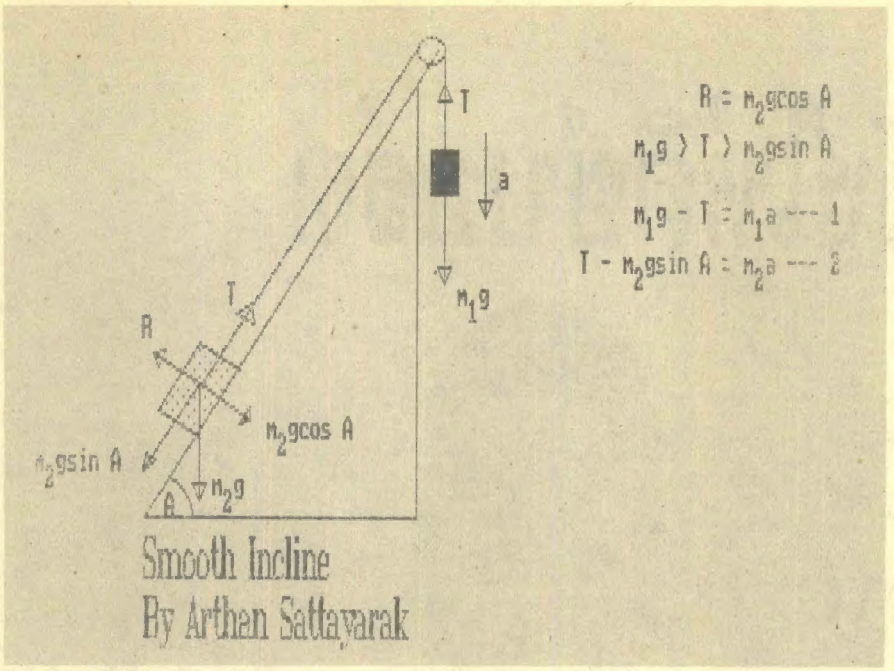
ชุดคำสั่ง Text เพื่อแสดงให้รู้ว่า แรงแต่ละอันสัมพันธ์กันอย่างไร ซึ่งชุดคำสั่งนี้เป็นการสั่งให้ปรากฏข้อความด้วยคำสั่ง OutTextXY คิดว่าแม้ไม่อธิบายการทำงานไว้ ผู้อ่านก็คงทำความเข้าใจได้ไม่ยาก



รูปที่ 2 ภาพแสดงรูปแบบของการระบายรูปทึบจากคำสั่ง SetFillStyle และคำสั่ง FloodFill

```
{$I INITGR.PAS}

PROCEDURE SmoothIncline;
begin
 moveto(150,250); lineto(350,50);
 lineto(350,250); lineto(150,250);
 bar(360,80,390,100);
 moveto(190,210); lineto(160,198);
 lineto(190,168); lineto(220,180);
 setfillstyle(10,1);  floodfill(190,200,blue);
 circle(365,30,10); line(350,50,365,30);
 line(195,188,358,25);   line(375,30,375,80);
 textshow(0,1,'h');  outtextxy(165,240,'A');
 arc(150,250,0,54,30);
end;

PROCEDURE Force(cc :integer);
begin
 setcolor(cc);
 line(375,101,375,140);  moveto(380,135);
 lineto(370,135); lineto(375,140);  lineto(380,135);
 outtextxy(385,140,'m g'); outtextxy(385,145,' 1');
 moveto(370,55); lineto(375,50);  lineto(380,55);
 lineto(370,55); outtextxy(384,52,'T');
 moveto(195,188);  lineto(195,240); lineto(190,235);
 lineto(200,235);  lineto(195,240);
 outtextxy(203,235,'m g'); outtextxy(203,240,' 2');
 moveto(235,148);  lineto(226,150); lineto(235,155);
 lineto(235,148);  outtextxy(224,140,'T');
 moveto(195,188); lineto(157,225);  lineto(158,218);
 lineto(166,222); lineto(157,225);
 outtextxy(80,220,'m gsin A'); outtextxy(80,225,' 2');
 moveto(195,188); lineto(234,207);  lineto(232,202);
 lineto(226,208); lineto(234,207);
 outtextxy(240,205,'m gcos A');    outtextxy(240,210,' 2');
 moveto(195,188); lineto(155,168); lineto(158,173);
 lineto(164,168); lineto(155,168); outtextxy(150,155,'R');
 moveto(400,70);  lineto(400,110); lineto(405,105);
 lineto(395,105); lineto(400,110); outtextxy(410,90,'a');
end;

PROCEDURE text;
begin
 outtextxy(450, 60,'           R = m gcos A');
 outtextxy(450, 65,'                2');
 outtextxy(450, 80,'     m g > T > m gsin A');
 outtextxy(450, 85,'      1         2');
 outtextxy(450,110,'    m g - T = m a --- 1');
 outtextxy(450,115,'      1         1 ');
 outtextxy(450,125,'T - m gsin A = m a --- 2');
 outtextxy(450,130,'     2          2');
end;

PROCEDURE Movepix;
var
 pix1,pix2   : pointer;
 size  : word;
 x :integer;
begin
 size := imageSize(360,75,390,100);   x:=0;
 getmem(pix1,size);   getimage(360,75,390,100,pix1^);
 size := imageSize(155,165,220,210);
 getmem(pix2,size);   getimage(155,165,220,210,pix2^);
 repeat
   x :=x+1;
   putimage(360,75+x,pix1^,0);    {normalput}
   putimage(155+x,165-x,pix2^,0);
 until x >100;
end;  { Movepix }

{++++++++++ (Smooth Incline :TURBO PASCAL V 4.0 ) ++++++++++}
begin
  Opengraph;
  Textshow(1,2,'h');
  outtextxy(150,270,'Smooth Incline');
  outtextxy(150,300,'By Arthan Sattayarak');
  SmoothIncline;       waittogo;
  Force(1);  text;     waittogo;
  Force(0);  Movepix;  waittogo;
  CloseGraph;
end.
```

รูปที่ 3 โปรแกรมแสดงการเคลื่อนที่ของวัตถุ 2 ชิ้น โดยชิ้นหนึ่งอยู่ในแนวดิ่ง และอีกชิ้นหนึ่งอยู่บนระนาบเอียง



รูปที่ 4 ภาพที่เกิดจากโปรแกรมในรูปที่ 3 ขณะที่วัตถุทั้งสองยังไม่เคลื่อนที่ แสดงแรงต่าง ๆ ที่กระทำต่อวัตถุทั้งสอง

เมื่อเราสั่งให้โปรแกรมทำงานตามโปรแกรมส่วนท้ายสุด เมื่อทำงานมาถึงคำสั่ง Text;ก็จะได้รูปดังรูปที่ 4 นั่นเอง ขอให้สังเกตการสั่งให้แสดง m_1g, m_2g ว่าสามารถกำหนดได้อย่างไร

ชุดคำสั่ง Movepix เพื่อแสดงการเคลื่อนที่ของวัตถุทั้งสองโดยวัตถุที่อยู่ในแนวดิ่งจะเคลื่อนที่ลง ส่วนวัตถุที่อยู่บนพื้นเอียงจะเคลื่อนที่ขึ้นตามพื้นเอียง อธิบายได้ดังนี้

1. กำหนดตัวแปร pix1, pix2 เพื่อใช้จำภาพของวัตถุทั้งสองไว้, size ตัวแปรเพื่อใช้กำหนดขอบเขตของภาพวัตถุทั้งสองในการจองเนื้อที่ในหน่วยความจำ, x ตัวแปรเพื่อใช้ในการเคลื่อนที่

โดยการนับบรรทัดถัดจากคำสั่ง begin จะได้การทำงานดังนี้

2. บรรทัดที่ 1-2 กำหนดขอบเขตของวัตถุที่ห้อยในแนวดิ่ง, จองเนื้อที่ในหน่วยความจำ และจำภาพของวัตถุนี้ในตัวแปร pix1 และกำหนดค่าตัวแปร x=0 ด้วย

3. บรรทัดที่ 3-4 กำหนดขอบเขตของวัตถุที่อยู่บนพื้นเอียง, จองเนื้อที่ในหน่วยความจำและจำภาพของวัตถุนี้ในตัวแปร pix2

4. ช่วง repeat/until x >100; เป็นการกำหนดค่าตัวแปร x ให้มีค่าเพิ่มขึ้นทีละ 1 พร้อมกับแสดงภาพของวัตถุที่ห้อยในแนวดิ่ง(pix1) ที่ตำแหน่งใหม่คือ 360,75+x และแสดงภาพวัตถุที่อยู่บนพื้นเอียง(pix2) ที่ตำแหน่งใหม่คือ 155 +x, 165-x ซึ่งจะทำให้เกิดภาพการเคลื่อนที่ของวัตถุทั้งสองจนกว่าค่าของตัวแปร x มากกว่า 100 จึงหยุดการเคลื่อนที่ สำหรับภาพการทำงานนั้นท่านต้องนั่งคีย์โปรแกรมเอง เมื่อรันดูก็จะรู้ว่าได้ภาพอย่างไร

ส่วนท้ายของโปรแกรมคือลำดับการทำงานโดยจะแสดงข้อความให้ทราบว่าเป็นเรื่องของพื้นเอียงลื่นและชื่อผู้เขียนตามด้วยการเรียกชุดคำสั่งต่าง ๆ ที่กล่าวมาแล้วข้างต้นมาทำงานให้เป็นไปตามที่ต้องการ

หวังว่าท่านผู้อ่านคงจะสามารถทำภาพเคลื่อนที่ตามแนวทางของเทอร์โบปาสคาล 4.0 โดยอาศัยแนวจากโปรแกรมครั้งนี้และคงได้ทราบคำสั่งด้านกราฟิกเพิ่มขึ้น คอยพบกับการเรียนรู้คำสั่งใหม่ ๆ และ CAI วิชาฟิสิกส์ได้ในฉบับต่อไป มีปัญหาที่ท่านสนใจ หรือมีข้อแนะนำเพิ่มเติม หรือมีข้อท้วงติง กรุณาติดต่อผู้เขียนได้ที่ ร.ร.ลำปางกัลยาณี อ.เมือง ลำปาง 52000 พบกันใหม่...ฉบับหน้าครับ

แก้ที่ผิด ในบทความเรื่อง พิมพ์ภาพกราฟิกด้วยปาสคาล 4.0 ในวารสารไมโครฯ 44 หน้า 195-198 ดังนี้

1. ESC 1 ทุกแห่งในบทความเรื่องนี้ แก้เป็น ESC l (อักษรแอลตัวเล็ก)
2. หน้า 198 คำสั่ง USES crt, printer, graph, thai, thaibig; นั้นให้แก้เป็น USES dos, crt, printer, graph, thai, thaibig;